102. (new) The computer program product according to claim 101, wherein the implemented method includes analyzing the server responses in order to extract attributes of said application interface elements.

REMARKS

Applicants respectfully request reconsideration of this application, as amended, and reconsideration of the Office Action dated March 19, 2004. Upon entry of this Amendment, Claims 6–102 will be pending in this application.

In the Office Action, the Examiner rejected Claims 1–5, under 35 U.S.C. § 102, as being anticipated by Reshef et al., U.S. Pat. No. 6,584,569 ("the Reshef Patent"). Without admitting that the Reshef Patent is valid prior art, Applicants request that Claims 1–5 be cancelled without prejudice.

New Claims 6–71 are presented to make certain that all aspects of Applicants' invention are adequately claimed. Applicants submit that these claims are supported by specification, as originally filed, and are in condition for allowance. Allowance of Claims 6–71 is earnestly solicited.

REQUEST FOR INTERFERENCE UNDER 37 C.F.R. § 1.607

I. Identification of Interfering Patent.

With respect to newly presented Claims 72-102, these claims have been copied from the Reshef Patent, U.S. Pat. No. 6,584,569, granted June 24, 2003. The following table shows the correspondence between Applicants' Claims 72 to 102 and the claims of the Reshef Patent:

Applicants' Claim	Reshef Claim	Applicants' Claim	Reshef Claim	Applicants' Claim	Reshef Claim
72	4 (incl. 1)	83	32	**94**	56
73	5	84	33	95	57
74	7	**85**	41	96	58
75	8	86	42	97	59
76	17 (incl. 13)	**87**	44	98	60
77	18	88	45	99	61
78	19	89	46	100	62
79	23	90	47	101	66
80	24	91	48	102	67
81	30	92	50		
82	31	93	51		

*Independent Claims in **Bold**

II. Application of Claim Terms.

In accordance with 37 C.F.R. § 1.607(a), the above-identified application, as originally filed, supports the copied claims. Pursuant to 37 C.F.R. § 1.607(a)(5), and in order to assist the Examiner, Applicants provide the following table indicating at least some of the support for the added claims. There may be additional support not set forth in the table.

Copied Claim	Applicants' Disclosure
Claim 72 (Reshef Claim 4, including subject matter of Reshef base Claim 1).	
A method for detecting security vulnerabilities in a web application executing on a web server or web application server,	Page 1, lines 6-8 Page 4, lines16-19

comprising:	
actuating the application in order to discover pre-defined elements of the application's interface with external clients;	Page 9, lines 1-19 Figure 2, steps 205, 210, and 215
generating client requests having unauthorized values for said elements in order to generate exploits unique to the application;	Page 15, lines 19-21 Page 16, lines 1-7 and 10-21 Page 17, lines 3-13 and 17-19 Figure 2, step 245
attacking the application using the exploits; and	Page 15, lines 19-21 Page 16, lines 1-7 and 10-21 Page 17, lines 3-13 and 19-20 Figure 2, step 245
evaluating the results of the attack;	Page 16, lines 7-9 Page 17, lines 1-2 and 20 Page 18, line 1 Figure 2, step 250
wherein actuating the application includes:	
sending an authorized client request in order to receive a server response;	Page 9, lines 1-6 Figure 2, step 205
parsing the response in order to discover links encapsulated therein; and	Page 9, lines 7-14 Figure 2, step 210
actuating discovered links in accordance with authorized client functionality in order to generate additional authorized client requests.	Page 9, lines 15-19 Figure 2, step 215
Claim 73 (Reshef 5). The method according to claim 72, further including comparing discovered links to a filter and not generating authorized client requests for links matching the filter.	Page 9, lines 20-23 Page 10, lines 1-2
Claim 74 (Reshef 7). The method according to claim 72, wherein said application interface elements are discovered by parsing at least one of the authorized client requests and server responses resulting therefrom.	Page 16, lines 6-7 and 20-21 Page 9 lines 4-14
Claim 75 (Reshef 8). The method according to claim 74, further including analyzing the server responses in order to extract attributes of said application interface elements.	Page 9, lines 12-14
Claim 76 (Reshef 17, including subject matter of Reshef base Claim 13).	

A method for detecting security vulnerabilities in a hypertext-based web application installed on a web server or web application server, comprising:	Page 1, lines 6-8 Page 4, lines 16-19
traversing the application in order to discover and actuate links therein;	Page 9, lines 1-19 Figure 2, steps 205, 210 and 215
analyzing messages that flow or would flow between an authorized client and the web server in order to discover elements of the application's interface with external clients and attributes of said elements;	Page 9, lines 1-19 Figure 2, steps 205, 210 and 215
generating unauthorized client requests in which said elements are mutated;	Page 15, lines 19-21 Page 16, lines 1-7 and 10-21 Page 17, lines 3-13 and 17-19 Figure 2, step 245
sending the mutated client requests to the server; and	Page 15, lines 19-21 Page 16, lines 1-7 and 10-21 Page 17, lines 3-13 and 19-20 Figure 2, step 245
receiving server responses to the unauthorized client requests and evaluating the results thereof;	Page 16, lines 7-9 Page 17, lines 1-2 and 20 Page 18, line 1 Figure 2, step 250
wherein traversing the application includes:	
sending an authorized client request in order to receive a server response;	Page 9, lines 1-6 Figure 2, step 205
parsing the response in order to discover links encapsulated therein; and	Page 9, lines 7-14 Figure 2, step 210
actuating discovered links in accordance with authorized client functionality in order to receive authorized server responses from which additional authorized client requests can be generated.	Page 9, lines 15-19 Figure 2, step 215
Claim 77 (Reshef 18). The method according to claim 76, further including comparing discovered links to a filter and not generating authorized client requests for links matching the filter.	Page 9, lines 20-23 Page 10, lines 1-2
Claim 78 (Reshef 19). The method according to claim 76, wherein, in the event the authorized client request requires user-interactive parameters, supplying pre-	Page 19, lines 13-18

configured values therefor.	
Claim 79 (Reshef 23). The method according to claim 76, wherein said application interface elements are discovered by parsing at least one of the authorized client requests and server responses resulting therefrom.	Page 16, lines 6-7 and 20-21 Page 9 lines 4-14
Claim 80 (Reshef 24). The method according to claim 79, further including analyzing the server responses in order to extract attributes of said application interface elements.	Page 9, lines 12-14
Claim 81 (Reshef 30). A scanner system, provided on a computer, for detecting security vulnerabilities in a HTML-based web application installed on a web server or web application server, the scanner system comprising:	Page 1, lines 6-8 Page 4, lines 16-19
a crawling engine for traversing the application in order to discover and actuate links therein;	Page 9, lines 1-19 Figure 2, steps 205, 210 and 215
an analysis engine for analyzing messages that flow or would flow between an authorized client and the web server in order to discover elements of the application's interface with external clients and attributes of said elements and for generating unauthorized client requests in which said elements are mutated; and	Page 9, lines 1-19 Figure 2, steps 205, 210 and 215
an attack engine for sending the mutated client requests to the server; receiving server responses to the unauthorized client requests and evaluating the results thereof.	Page 15, lines 15-21 Page 16, lines 1-21 Page 17, lines 1-20 Page 18, lines 1-6 Figure 2, steps 245 and 250
Claim 82 (Reshef 31). The scanner system according to claim 81, wherein the crawling engine:	
sends an authorized client request in order to receive a server response;	Page 9, lines 1-6 Figure 2, step 205
invokes the parsing engine to parse the	Page 9, lines 7-14

response in order to discover links encapsulated therein; and	Figure 2, step 210
actuates discovered links in accordance with authorized client functionality in order to receive authorized server responses from which additional authorized client requests can be generated.	Page 9, lines 15-19 Figure 2, step 215
Claim 83 (Reshef 32). The scanner system according to claim 82, wherein the crawling engine compares discovered links to a filter and does not generate authorized client requests for filtered links.	Page 9, lines 20-23 Page 10, lines 1-2
Claim 84 (Reshef 33). The scanner system according to claim 82, wherein, in the event the authorized client request requires user-interactive parameters, the crawling engine supplies pre-configured values therefor.	Page 19, lines 13-18
Claim 85 (Reshef 41). A crawling engine, provided on a computer, for automatically traversing a hypertext-based web site, comprising:	Page 1, lines 6-8 Page 4, lines 16-19 and 22-23 Page 5, lines 1-2 and 10-12 Page 6, lines 6-8 Page 7, lines 20-23 Page 8, lines 1-3 and 18-22
means for sending a client request in order to receive a server response;	Page 9, lines 1-6 Figure 2, step 205
means for parsing the response in order to discover links encapsulated therein;	Page 9, lines 7-14 Figure 2, step 210
means for actuating one or more discovered links in accordance with authorized client functionality in order to receive one or more server responses from which one or more additional client requests are generated; and	Page 9, lines 15-19 Figure 2, step 215
means for automatically supplying values to user-interactive parameters in the additional client requests, if required.	Page 19, lines 13-18
Claim 86 (Reshef 42). The engine according to claim 85, further including means for comparing discovered links to a filter and not generating client requests for filtered links.	Page 9, lines 20-23 Page 10, lines 1-2

Claim 87 (Reshef 44). A computer program product comprising a computer readable medium having computer readable code embodied therein, the computer readable code, when executed, causing a computer to implement a method for detecting security vulnerabilities in a web application executing on a web server or web application server, comprising:	Page 7, lines 20-23 Page 8, lines 1-3
actuating the application in order to discover pre-defined elements of the application's interface with external clients;	Page 9, lines 1-19 Figure 2, steps 205, 210 and 215
generating client requests having unauthorized values for said elements in order to generate exploits unique to the application;	Page 15, lines 19-21 Page 16, lines 1-7 and 10-21 Page 17, lines 3-13 and 17-19 Figure 2, step 245
attacking the application using the exploits; and	Page 15, lines 19-21 Page 16, lines 1-7 and 10-21 Page 17, lines 3-13 and 19-20 Figure 2, step 245
evaluating the results of the attack.	Page 16, lines 7-9 Page 17, lines 1-2 and 20 Page 18, line 1 Figure 2, step 250
Claim 88 (Reshef 45). The computer program product according to claim 87, wherein an application interface element is a path parameter.	Page 17, lines 3-13 Page 18, lines 7-23
Claim 89 (Reshef 46). The computer program product according to claim 87, wherein an application interface element is a data parameter.	Page 17, lines 17-20 Page 18, lines 1-3 Page 19, lines 1-11
Claim 90 (Reshef 47). The computer program product according to claim 87, wherein, in the implemented method, actuating the application includes:	
sending an authorized client request in order to receive a server response;	Page 9, lines 1-6 Figure 2, step 205
parsing the response in order to discover links encapsulated therein; and	Page 9, lines 7-14 Figure 2, step 210
actuating discovered links in accordance with	Page 9, lines 15-19

authorized client functionality in order to generate additional authorized client requests.	Figure 2, step 215
Claim 91 (Reshef 48). The computer program product according to claim 90, wherein the implemented method includes	
comparing discovered links to a filter and not generating authorized client requests for links matching the filter.	Page 9, lines 20-23 Page 10, lines 1-2
Claim 92 (Reshef 50). The computer program product according to claim 90, wherein, in the implemented method, said application interface elements are discovered by parsing at least one of the authorized client requests and server responses resulting therefrom.	Page 16, lines 6-7 and 20-21 Page 9 lines 4-14
Claim 93 (Reshef 51). The computer program product according to claim 92, wherein the implemented method includes analyzing the server responses in order to extract attributes of said application interface elements.	Page 9, lines 12-14
Claim 94 (Reshef 56). A computer program product comprising a computer readable medium having computer readable code embodied therein, the computer readable code, when executed, causing a computer to implement a method for detecting security vulnerabilities in a hypertext-based web application installed on a web server or web application server, comprising:	Page 7, lines 20-23 Page 8, lines 1-3
traversing the application in order to discover and actuate links therein;	Page 9, lines 1-19 Figure 2, steps 205, 210 and 215
analyzing messages that flow or would flow between an authorized client and the web server in order to discover elements of the application's interface with external clients and attributes of said elements;	Page 9, lines 1-19 Figure 2, steps 205, 210 and 215
generating unauthorized client requests in which said elements are mutated;	Page 15, lines 19-21 Page 16, lines 1-7 and 10-21 Page 17, lines 3-13 and 17-19

	Figure 2, step 245
sending the mutated client requests to the server; and	Page 15, lines 19-21 Page 16, lines 1-7 and 10-21 Page 17, lines 3-13 and 19-20 Figure 2, step 245
receiving server responses to the unauthorized client requests and evaluating the results thereof.	Page 16, lines 7-9 Page 17, lines 1-2 and 20 Page 18, line 1 Figure 2, step 250
Claim 95 (Reshef 57). The computer program product according to claim 94, wherein an application interface element is a path parameter.	Page 17, lines 3-13 Page 18, lines 7-23
Claim 96 (Reshef 58). The computer program product according to claim 94, wherein an application interface element is a data parameter.	Page 17, lines 17-20 Page 18, lines 1-3 Page 19, lines 1-11
Claim 97 (Reshef 59). The computer program product according to claim 94, wherein an application interface element is a cookie.	Page 18, lines 7-23
Claim 98 (Reshef 60). The computer program product according to claim 94, wherein, in the implemented method, traversing the application includes:	
sending an authorized client request in order to receive a server response;	Page 9, lines 1-6 Figure 2, step 205
parsing the response in order to discover links encapsulated therein; and	Page 9, lines 7-14 Figure 2, step 210
actuating discovered links in accordance with authorized client functionality in order to receive authorized server responses from which additional authorized client requests can be generated.	Page 9, lines 15-19 Figure 2, step 215
Claim 99 (Reshef 61). The computer program product according to claim 98, wherein the implemented method includes comparing discovered links to a filter and not generating authorized client requests for links	Page 9, lines 20-23 Page 10, lines 1-2

matching the filter.	
Claim 100 (Reshef 62). The computer program product according to claim 98, wherein, in the implemented method, in the event the authorized client request requires user-interactive parameters, supplying pre-configured values therefor.	Page 19, lines 13-18
Claim 101 (Reshef 66). The computer program product according to claim 98, wherein, in the implemented method, said application interface elements are discovered by parsing at least one of the authorized client requests and server responses resulting therefrom.	Page 16, lines 6-7 and 20-21 Page 9 lines 4-14
Claim 102 (Reshef 67). The computer program product according to claim 101, wherein the implemented method includes analyzing the server responses in order to extract attributes of said application interface elements.	Page 9, lines 12-14

III. Presentation of Proposed Counts.

Pursuant to 37 C.F.R. § 1.607(a)(2), Applicants "present" the following proposed Counts:

1. A method for detecting security vulnerabilities in a web application executing on a web server or web application server, comprising:

actuating the application in order to discover pre-defined elements of the application's interface with external clients;

generating client requests having unauthorized values for said elements in order to generate exploits unique to the application;

attacking the application using the exploits; and evaluating the results of the attack;

wherein actuating the application includes:

sending an authorized client request in order to receive a server response;

parsing the response in order to discover links encapsulated therein; and

actuating discovered links in accordance with authorized client functionality in order to generate additional authorized client requests.

2. A method for detecting security vulnerabilities in a hypertext-based web application installed on a web server or web application server, comprising:

traversing the application in order to discover and actuate links therein;

analyzing messages that flow or would flow between an authorized client and the web server in order to discover elements of the application's interface with external clients and attributes of said elements;

generating unauthorized client requests in which said elements are mutated;

sending the mutated client requests to the server; and

receiving server responses to the unauthorized client requests and evaluating the results thereof;

wherein traversing the application includes:

sending an authorized client request in order to receive a server response;

parsing the response in order to discover links encapsulated therein; and

actuating discovered links in accordance with authorized client functionality in order to receive authorized server responses from which additional authorized client requests can be generated.

3. A scanner system, provided on a computer, for detecting security vulnerabilities in a HTML-based web application installed on a web server or web application server, the scanner system comprising:

a crawling engine for traversing the application in order to discover and actuate links therein;

an analysis engine for analyzing messages that flow or would flow between an authorized client and the web server in order to discover elements of the application's interface with external clients and attributes of said elements and for generating unauthorized client requests in which said elements are mutated; and

an attack engine for sending the mutated client requests to the server; receiving server responses to the unauthorized client requests and evaluating the results thereof.

4. A crawling engine, provided on a computer, for automatically traversing a hypertext-based web site, comprising:

means for sending a client request in order to receive a server response;

means for parsing the response in order to discover links encapsulated therein;

means for actuating one or more discovered links in accordance with authorized client functionality in order to receive one or more server responses from which one or more additional client requests are generated; and

means for automatically supplying values to user-interactive parameters in the additional client requests, if required.

5. A computer program product comprising a computer readable medium having computer readable code embodied therein, the computer readable code, when executed, causing a computer to implement a method for detecting security vulnerabilities in a web application executing on a web server or web application server, comprising:

actuating the application in order to discover pre-defined elements of the application's interface with external clients;

generating client requests having unauthorized values for said elements in order to generate exploits unique to the application;

attacking the application using the exploits; and

evaluating the results of the attack.

6. A computer program product comprising a computer readable medium having computer readable code embodied therein, the computer readable code, when executed, causing a computer to implement a method for detecting security vulnerabilities in a hypertext-based web application installed on a web server or web application server, comprising:

traversing the application in order to discover and actuate links therein;

analyzing messages that flow or would flow between an authorized client and the web server in order to discover elements of the application's interface with external clients and attributes of said elements;

generating unauthorized client requests in which said elements are mutated;

sending the mutated client requests to the server; and receiving server responses to the unauthorized client requests and evaluating the results thereof.

IV. Identification of Corresponding Claims.

Pursuant to 37 C.F.R. § 1.607(a)(3)–(4), Applicants submit the following table showing the correspondence between Applicants' claims and the claims of the Reshef Patent to the proposed counts:

Proposed Count	Applicants' Corresponding Claim(s)	Corresponding Claim(s) of Reshef Patent
1	72	4
2	76	17
3	81	30
4	85	41
5	87	44
6	94	56

Applicants submit that application Claims 6–71, 73–75, 77–80, 82–84, 86, 88–93, and 95–102 do not correspond to proposed Counts 1–6. Additionally, Applicants concede that Claims 1–3, 5–16, 18–29, 31–40, 42–43, 45–55, and 57–72 of the Reshef Patent do not correspond to proposed Counts 1–6. It is respectfully submitted that Applicants' Claims 6–71, 73–75, 77–80, 82–84, 86, 88–93, and 95–102 and Claims 1–3, 5–16, 18–29, 31–40, 42–43, 45–55, and 57–72 of the Reshef Patent define "separate patentable inventions," within the meaning of 35 C.F.R. § 1.601(n), from the claims corresponding to the proposed counts.

V. Applicants' Earlier Effective Filing Date.

Applicants respectfully note that the above-identified application has an earlier filing date than the application which issued as the Reshef Patent. Although the Reshef Patent's Provisional application, Serial No. 60/186,892 filed on March 3, 2000 ("the Reshef

Provisional"), has an earlier filing date than the above-identified application, Applicants submit that they have the earliest "effective filing date," within the meaning of 37 C.F.R. § 1.601(g), with respect to the subject matter of copied Claims 72–102 and proposed Counts 1–6 because the Reshef Patent should not be accorded the benefit of priority of the filing date of the Reshef Provisional as to such subject matter.

Specifically, Applicants submit that the Reshef Provisional lacks disclosure for:

(1) the recursive functionality claimed by copied independent Claims 72, 76, and 85, and corresponding to proposed Counts 1, 2, and 4;

(2) the automatic functionality claimed by copied independent Claims 81, 85, 87, and 94, and corresponding to proposed Counts 3, 4, 5, and 6; and

(3) the "means for automatically supplying values to user-interactive parameters in the additional client requests" claimed by copied independent Claim 85 and corresponding to Count 4.

(The Examiner will note that copied Claim 85 corresponding to proposed Count 4 claims all of the aforementioned functionalities).

Moreover, Applicants submit that the Reshef Provisional does not disclose the particular embodiments of the copied claims dependent therefrom.

(1) Counts 1, 2, and 4 – The Recursive Functionality.

With respect to proposed Counts 1, 2, and 4, these counts, and the claims corresponding to these counts, claim a process or system wherein the information obtained during an initial examination of a web application is used to further analyze the web application at a deeper level, i.e., a process or system wherein the security analysis is performed recursively. For example, proposed Counts 1 and 2 define actuating or traversing the application to include "sending an authorized client request in order to receive a server response[,] parsing the response in order to discover links encapsulated therein[,] *and actuating discovered links* in accordance with authorized client functionality *in order to generate additional authorized client requests*." (emphasis supplied). Likewise, proposed Count 4 claims a "means for *actuating one or more discovered links* in accordance with authorized client functionality in order to receive one or more server responses *from which one or more additional client requests are generated*." (emphasis supplied).

a. <u>The Reshef Patent Disclosure of Counts 1, 2, and 4.</u>

To illustrate, the specification of the Reshef Patent describes the "crawling" stage of the scanning operation to be initiated by providing a starting or root Uniform Resource Locator ("URL") to the scanner (Col. 5, lines 61-62). The starting URL is stored in a "WorkQueue" data structure (Col. 5, lines 62-63). The starting URL is then retrieved from WorkQueue (Col. 6, line 1) and transmitted as an HTTP request to the web server (Col. 6, lines 26-28). The response from the web server is received (Col. 6, lines 31-33) and parsed to extract links encapsulated therein (Col. 6, lines 42-44). Each extracted link is stored in WorkQueue (Col. 6, lines 44-46). As shown in Fig. 3A, the process loops back (from step 124 to step 112), and each extracted link will be retrieved from WorkQueue and parsed for additional links. Overall, a recursive process for analyzing a web application is described.

b. <u>Applicants' Disclosure of Counts 1, 2, and 4.</u>

Applicants submit that the above-identified application discloses the same recursive process, indeed, in a more straightforward manner than the Reshef Patent. Applicants' specification explicitly states "[i]n yet another aspect of the invention, an Internet security analysis process is recursive, gathering information of security vulnerabilities and then exploiting that information to search for additional security vulnerabilities." (Page 5, lines 13-16). Similarly, the specification discloses that "the Internet security analysis system further exploits each of the vulnerabilities detected in the step 235 and added to the security vulnerability database to search for further security vulnerabilities and to gather other information regarding the target Web site. This process is recursively repeated until no new data is obtained." (Page 15, lines 8-14).

More particularly, the above-identified application discloses first retrieving a default Web page for a target web site (Page 9, lines 4-6; Fig. 2, step 205). Next, the system parses the default Web page to search for any linked-to Web pages or other objects which are referenced, and the URL of each linked-to Web page is stored in a database (Page 9, lines 7-12; Fig. 2, step 210). Each linked-to Web page is then in turn retrieved from the database and parsed to search for any further linked-to Web pages (Page 9, lines 15-18; Fig. 2, step 215). The process is repeated for each link and each linked-to Web page that is found (Page 9, lines 18-19; Fig. 2, loop from step 215 to step 210). "Together, the steps . . . constitute a 'Webcrawl' of the entire target Web site, performed by parsing through each retrieved Web page in turn to find additional links." (Page 11, lines 7-9).

c. **The Reshef Provisional Fails to Disclose Counts 1, 2, and 4.**

The Reshef Provisional, on the other hand, does not teach such a recursive process. The first stage of the operation described by the Reshef Provisional consists of "the user us[ing] his favorite browser to browse his site, [wherein the system] logs and performs relevant data extraction (HTTP + HTML) on all requests and responses and stores the output in the database. The request and the response are also stored in a database." (Page 3, lines 14-19). There is no disclosure of using the extracted data to further analyze the web application. With reference to the language of proposed Counts 1, 2, and 4, there is no disclosure of actuating a link that was "discovered" during the parsing of a response and no disclosure of generating "additional" requests.

Consequently, the Reshef Patent should not be accorded the benefit of priority of the filing date of the Reshef Provisional as to the subject matter of proposed Counts 1, 2, and 4, and corresponding Claims 72, 76, and 85. Thus, because the above-identified application was filed before the application which issued as the Reshef Patent, Applicants should be deemed to have the earlier "effective filing date," within the meaning of 37 C.F.R. § 1.601(g), with respect to proposed Counts 1, 2, and 4.

(2) **Counts 3, 4, 5, and 6 – The Automatic Functionality.**

With respect to proposed Counts 3, 4, 5, and 6, these counts, and the claims corresponding to these counts, claim a system wherein one or more of the steps for analyzing a web application for security vulnerabilities is performed automatically using software routines, i.e., performed without manual intervention. For example, proposed Count 3 claims "*a crawling engine* for traversing the application in order to discover and actuate links therein." (emphasis supplied). Likewise, proposed Count 4 claims "*[a] crawling engine*, provided on a computer, for *automatically* traversing a hypertext-based web site." (emphasis supplied). Finally, proposed Counts 5 and 6 claim a computer program having code that "*caus[es] a computer to implement a method* for detecting security vulnerabilities" in a web application, the method comprising "traversing" or "actuating" the application. (emphasis supplied).

a. The Reshef Patent Disclosure of Counts 3, 4, 5, and 6.

To illustrate, the specification of the Reshef Patent plainly states, "[t]he invention seeks to provide a scanner for automatically detecting potential application-level vulnerabilities or security flaws in a web application." (Col. 2, lines 16-18). Further, the Reshef Patent discloses that although each stage of the operation is preferably initiated manually, the stages "could be automatically actuated if desired." (Col. 3, lines 41-43).

Specifically, the Reshef Patent describes, in significant detail, the automated functionality of the disclosed "crawling engine." (*See* Col. 4, lines 39-60; Col. 5, lines 44-50). The disclosed crawling engine is described as being capable of operating in either an automatic, manual, or interactive mode (Col. 4, lines 40-42). "In the automated mode, the crawling engine 13 automatically scans the whole web application or site and discovers all the links or URL's associated therewith." (Col. 5, lines 45-48).

b. Applicants' Disclosure of Counts 3, 4, 5, and 6.

Applicants submit that the above-identified application discloses the same automated functionality. Applicants' specification explicitly states, "[i]n a preferred embodiment, the Internet security analysis system 100 executes one or more software routines to perform an automatic security analysis process. The automatic security analysis process analyzes a target Web site to identify security flaws." (Page 7, line 20 to Page 8, line 1). Figure 2 of Applicants' disclosure is described as "a flowchart for a preferred embodiment process 200 of automatically parsing through a target Web site to identify possible security holes[,] [wherein] [t]he process 200 may be initiated by a user and executed by a software program running on the Internet security analysis system . . . ." (Page 8, lines 16-20). In fact, the above-identified application specifically refers to the portion of the system responsible for crawling the application as a "Webcrawl engine." (Page, 14, lines 16-17).

c. The Reshef Provisional Fails to Disclose Counts 3, 4, 5, and 6.

The Reshef Provisional, on the other hand, does not disclose such automatic functionality. Indeed, the word "automated" or "automatic" does not appear in the Reshef Provisional. Almost every step of described method requires user intervention. For example, "[d]uring stage 1, *the user* uses his favorite browser to browse his site . . . ." (Page 3, lines 14-15) (emphasis supplied). "During stage 3, *the user* is presented with a report . . . which can be categorized and sorted." (Page 4, lines 8-10) (emphasis supplied). "In Stage 4, *the user* attempts the various exploits . . . ." (Page 4, line 11) (emphasis supplied). "In Stage 5, *the user* may request to create a report . . . ." (Page 4, line 16) (emphasis supplied). Finally, in Stage 6, "*the user* manually performs the requests and examines the results." (Page 4, lines 20-21) (emphasis supplied).

Although the Reshef Provisional makes reference to a "crawling" or data acquisition *state* (Page 6, lines 16 and 18), such language simply refers to a particular stage within the operation rather than to any automatic crawling functionality. There is no mention in the Reshef Provisional of the "crawling engine" disclosed in the Reshef Patent. To be sure, while Figure 2A of the Reshef Patent includes a "Crawl Engine" as part of the system architecture, Figure 2 of the Reshef Provisional shows no such component.

With reference to the language of proposed Counts 3 and 4, there is no disclosure in the Reshef Provisional of a "crawling engine" for automatically traversing an application. Similarly, with reference to the language of proposed Counts 5 and 6, the Reshef Provisional requires a user to implement the disclosed method and does not support "causing a computer to implement a method" for detecting security vulnerabilities.

Consequently, the Reshef Patent should not be accorded the benefit of priority of the filing date of the Reshef Provisional as to the subject matter of proposed Counts 3, 4, 5, and 6, and corresponding Claims 81, 85, 87, and 94. Thus, because the above-identified application was filed before the application which issued as the Reshef Patent, Applicants should be deemed to have the earlier "effective filing date," within the meaning of 37 C.F.R. § 1.601(g), with respect to proposed Counts 3, 4, 5, and 6.

(3) **Count 4 – The Form Fill-In Functionality.**

Finally, with reference to proposed Count 4, that count, and the claims corresponding to that count, claim a crawling engine having "means for automatically supplying values to user-interactive parameters in the *additional* client requests, if required." (emphasis supplied). The Reshef Patent describes the disclosed crawling engine as being capable of automatically filling in an HTML form using a pre-defined value when such a form is encountered while traversing the links of the target web site (Col. 4, lines 45-48). Likewise, Applicants' specification discloses a system that supports passing user-defined usernames and passwords to a login form if needed to gain access to the target web site, and that supports passing multiple usernames and passwords to test a login form by "brute force." (Page 19, lines 1-18). The Reshef Provisional, however, does not disclose such functionality.

Further, as described above, the Reshef Provisional does not disclose a recursive process which would generate the "additional" client requests required by the proposed count.

Accordingly, Applicants have presented another basis for determining that they have the earlier "effective filing date," within the meaning of 37 C.F.R. § 1.601(g), with respect to proposed Count 4.

VI. Inapplicability of 35 U.S.C. § 135(b).

Pursuant to 37 C.F.R. § 1.607(6), Applicants submit that the requirements of 35 U.S.C. § 135(b) are met. Because this Amendment is being filed prior to one year from June 24, 2003, the date on which the Reshef Patent was granted, 35 U.S.C. § 135(b)(1) is not applicable. Further, because the above-identified application (filed on November 28, 2000) was filed before the publication date (January 24, 2002) of the application which issued as the Reshef Patent, 35 U.S.C. § 135(b)(2) is not applicable. *See also* MPEP § 2307.02, Form Paragraph 23.14.01, Examiner's Note 2.

VII. Identification of Related Patents.

Without seeking to involve the above-requested interference with a third application or patent, Applicants submit that the Examiner should be aware of Appl. No. 10/393,497. Appl. No. 10/393,497 was filed on March 20, 2003 as a continuation of the Reshef Patent and published on December 18, 2003, Pub. No. US 2003/0233581 ("the Reshef Continuation"). Claim 1 of the Reshef Continuation, the only claim in the published application, is identical to Claim 1 of the Reshef Patent and does not correspond to proposed Counts 1–6. Nonetheless, because the claims of the Reshef Continuation may have been amended since publication so as not to be identical to Claim 1 of the Reshef Patent, Applicants' respectfully request that the Examiner determine whether such claims, if amended, correspond to any of the proposed counts.

VIII. Conclusion.

Pursuant to 37 C.F.R. § 1.607, and for the foregoing reasons, Applicants request that the Examiner propose an interference between the above-identified application and U.S. Pat. No. 6,584,569.

Additional claim fees for seventy-seven (77) total claims in excess of twenty (20) and for nine (9) independent claims in excess of three (3) are filed concurrently herewith as a result of this Amendment. If any additional fees are due in connection with the filing of this Amendment or the accompanying papers, such as fees under 37 C.F.R. §§ 1.16 or 1.17, please charge the fees to SGR Deposit Account No. 02-4300, Order No. 042600.005. If an additional extension of time under 37 C.F.R. § 1.136 is necessary that is not accounted for in the papers filed herewith, such an extension is requested. The additional extension fee also should be charged to SGR Deposit Account No. 02-4300, Order No. 042600.005. Any overpayment can be credited to Deposit Account No. 02-4300, Order No. 042600.005.

Respectfully submitted,

Dale Lischer

Dale Lischer, Reg. No. 28,438

Dated: June 18, 2004
SMITH, GAMBRELL & RUSSELL, LLP
1230 Peachtree Street, N.E.
Suite 3100, Promenade II
Atlanta, GA 30309-3592
TEL: (404) 815-3741
FAX: (404) 685-7041